WHAT IS CLAIMED IS:

1. A method of tracing the execution of a computer program comprising:

generating trace control information which specifies triggers and associated actions;

tracing executing of the computer program according to the trace control information, such that when one of said triggers occurs the corresponding action is performed; and

generating a trace log of said tracing, wherein the trace log reflects said actions performed during tracing.

2. The method of Claim 1, wherein said triggers include at least one of the following: the execution of the assembly code generated from a function entry, a function exit, or a source line; the activation of a software exception; the start or termination, normal or erroneous, of a process; and a user action.

3. The method of Claim 1, wherein a trigger is activated every predefined interval of time.

4. The method of Claim 1, wherein at least some of said actions are activated only if a plurality of conditions are satisfied while said triggers are activated, said conditions including comparing values of data passed in said execution to other values.

5. The method of Claim 1, wherein said actions include at least one of the following: writing to said trace log the stack dump of functions active at a time of a trigger; saving said trace log to a file; writing a comment to said trace log; stopping said tracing of said execution; and resuming said tracing of said execution.

6. A method of remotely debugging a client program which runs at a client site which is located remotely from a developer site, the method comprising:

using a first code module at the developer site to select a plurality of source code elements of the client program to be traced, and to generate trace control information which indicates said source code elements, wherein said trace control

information includes one or more pairs of triggers and actions, said one or more pairs of triggers and actions each specifying an event and an action to take in response to said event;

transmitting said trace control information to the client site;

at the client site, executing the client program together with a second code module which traces the execution of said client program based on said trace control information, to thereby generate a trace log which reflects execution of the client program;

transmitting said trace log from said client site to the developer site; and

at the developer site, using the trace log to debug the client program;

7. A software system that facilitates the process of identifying and isolating bugs within a client program without requiring modifications to the executable and source code files of the client program, the client program including at least a source code representation, an executable code representation, and debug information that links the source code representation to the executable code representation, the debug information generated by a compiler program during compilation of the client source code representation, the software system comprising:

a first code module that displays source code elements of said client on a display screen together with controls that enable a software developer to interactively specify one or more elements to be traced, the first code module configured to generate trace control information based on selections by said developer of said source code elements to be traced, said first code module using at least said debug information to generate said trace control information, said trace control information comprising conditional tracing data expressed as triggers and actions related to said triggers;

a second code module that attaches to a memory image of said object code representation of said client program based on said trace control information, said second code module configured to monitor execution of said client program and to generate, based at least in part on said triggers and said actions, trace information

that reflects said execution, said second code module configured to run in a same context as said client program; and

a third code module that translates said trace information into a human-readable form based on at least said debug information, and displays translated trace information on said display screen to allow said developer to analyze the execution of said client program.

8. A software execution tracing system for tracing a client program having at least a source code representation and an executable code representation, said software system comprising:

a first code module that attaches to a memory image of said client program, said first code module configured to monitor execution of said client program and to generate trace information that reflects said execution, said trace information generated at least in part by specified actions in response to specified triggers; and

a second code module that translates said trace information into a human-readable form, and displays translated trace information on said display screen to allow said developer to analyze the execution of said client program.

9. The software system of Claim 8, wherein the first and second code modules are configured to run under the control of a multi-processing operating system, and at least said first code module runs in a process memory space that has been allocated to said client program by said operating system, said first code module thereby tracing the execution of said client program without requiring context switches.

10. The software system of Claim 8, wherein said first code module is adapted to be sent to a remote user site together with trace control information, said trace control information including data describing said triggers and said actions, and where said first code module is adapted to be used at said remote user site as a stand-alone tracing component that enables a remote customer who does not have access to said source code representation or said second code module, to generate a trace file that represents execution of said client application at said remote site.

11. The software system of Claim 10, wherein said trace control information does not reveal source code element of said client program.

12. The software system of Claim 8, wherein said second code module displays said translated trace information on said display screen during execution of said client program.

13. The software system of Claim 8, wherein said second code module provides an offline analysis mode which provides functionality for interactively analyzing said trace information after the monitoring of said client application has completed.

14. The software system of Claim 8, wherein said second code module translates said trace information into a human-readable form based on at least build information, where said build information links said source code representation to said executable code representation

15. The software system of Claim 14, wherein said build information is generated by a build procedure.

16. The software system of Claim 14, wherein said build information comprises debug information generated by a compiler program during compilation of said client source code representation.

17. The software system of Claim 8, wherein said second code module displays process information, source code information, and trace detail information, said trace detail information comprising information regarding events specified by said triggers.

18. The software system of Claim 8, further comprising trace control information stored in a trace control file, said trace control information including specification of at least one trigger and at least one action to be taken in response to said at least one trigger.

19. The software system of Claim 18, wherein said trace control file is moved to a remote computer and used with said second code module and said client program on said remote computer.

20. The software system of Claim 18, wherein said at least one trigger includes

an occurrence of an event corresponding to at least one of: a function entry, a function exit, execution of a source line; the activation of a software exception; the start or termination, normal or erroneous, of a process; and a user action.

21. The software system of Claim 19, wherein said at least one action includes at least one of: writing a trace log comprising a stack dump of functions active at a time of said at least one trigger; saving a trace log to a file; writing a comment to said trace log; suspending tracing; and resuming tracing.

22. The software system of Claim 19, said at least one trigger including at least one condition, said condition specifying whether said trigger causes the execution of said at least one action.

23. The software system of Claim 8, wherein said condition comprises a logical expression based on at least one of: a constant, an address of a variable, a variable, a field of a variable, and a function return value.

24. The software system of Claim 23, wherein said first code module is further configured to collect trace log information based on conditional trace control information.

25. A method for simultaneously tracing the execution path of one or more client programs, said method comprising the steps of:

loading a client program into a computer memory to create an in-memory image of said client program;

instrumenting said in-memory image of said client program by attaching a trace library module to said client program;

collecting trace data relating to the execution of said client program and transferring said trace data to a trace log, wherein collecting trace data comprises taking actions in response to triggers;

transferring said encoded trace log to computer memory accessible by a trace analysis tool; and

analyzing said trace data using said executable analysis tool by decoding and displaying said trace data.

26. The method of Claim 25, further comprising the step of creating trace

control information, said trace control information comprising commands, triggers, and actions, used by said trace library module when collecting said trace data.

27. The method of Claim 26, wherein said trace control dataset comprises function addresses.

28. The method of Claim 25, wherein said trace log is stored in shared memory.

29. The method of Claim 25, wherein said library module is configured to run in a same context as said client.

30. The method of Claim 25, wherein said step of analyzing comprises translating said trace information into a human-readable form and showing events corresponding to said actions in a human-readable form.

31. The method of Claim 25, wherein said step of analyzing comprises translating said trace data into human-readable data based on at least build information and displaying said human-readable data on a display, where said build information links a source code representation to addresses in said trace control dataset.

32. The method of Claim 31, wherein said display comprises an execution call tree display and a source code display.

33. The method of Claim 32, wherein said execution call tree display and said source code display are synchronized such that said source code display displays source code for a function selected in said execution call tree.

34. The method of Claim 32, wherein said execution call tree display shows execution call trees for a plurality of processes.

35. The method of Claim 32, wherein said execution call tree display shows execution call trees for a plurality of threads.

36. The method of Claim 32, further comprising a trace tree showing results of actions taken in response to triggers.